In the claims:

For the Examiner's convenience, all pending claims are presented below with changes shown in accordance with the mandatory amendment format.

1. (Cancelled)

2. (Currently Amended) A computer-implemented method, comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

assigning an alias-node for one or more aliases representing an equivalence class of memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; and

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs.

3. (Currently Amended) The computer-implemented method of claim 2, further comprising performing a memory alias analysis of the intermediate language program to partition the memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class.

4. (Currently Amended) The computer-implemented method of claim 2, further comprising performing a program analysis using the dependence flow graph.

5. (Currently Amended) The computer-implemented method of claim 4, wherein the program analysis comprises for each alias-node in the dependence flow graph assigning an initial value to the alias corresponding to the alias-node and adding the alias-node to a set of nodes.

6. (Currently Amended) The computer-implemented method of claim 5, wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

7. (Cancelled)

8. (Previously Presented) A machine-readable medium that provides instructions, which when executed by a processor, cause the processor to perform operations comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

assigning an alias-node for one or more aliases representing an equivalence class of memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; and

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs.

9. (Previously Presented) The machine-readable medium of claim 8, wherein the operations further comprise performing a memory alias analysis of the intermediate language program to partition the memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class.

10. (Previously Presented) The machine-readable medium of claim 8, wherein the operations further comprise performing a program analysis using the dependence flow graph.

11. (Previously Presented) The machine-readable medium of claim 10, wherein the program analysis comprises for each alias-node in the dependence flow graph

assigning an initial value to the alias corresponding to the alias-node and adding the alias-node to a set of nodes.

12. (Previously Presented) The machine-readable medium of claim 11, wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

13. (Cancelled)

14. (Previously Presented) An apparatus, comprising:

a memory;

a processor coupled to the memory and having a set of instructions which when executed by the processor cause the processor to perform operations comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

assigning an alias-node for one or more aliases representing an equivalence class of memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; and

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs.

15. (Previously Presented) The apparatus of claim 14, wherein the operations further comprise performing a memory alias analysis of the intermediate language program to partition the memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class.

16. (Previously Presented) The apparatus of claim 14, wherein the operations further comprise performing a program analysis using the dependence flow graph.

17. (Previously Presented) The apparatus of claim 16, wherein the program analysis comprises for each alias-node in the dependence flow graph assigning an initial value to the alias corresponding to said alias-node and adding the alias-node to a set of nodes.

18. (Previously Presented) The apparatus of claim 17, wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

19.-20. (Cancelled)

21. (Previously Presented) An apparatus, comprising:

means for assigning a definition-node for one or more definition statements in a intermediate language program;

means for assigning a use-node for one or more use statements in the intermediate language program;

means for assigning an alias-node for one or more aliases representing an equivalence class of memory accesses;

means for introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; and

means for introducing an edge into the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs.

22. (Previously Presented) The apparatus of claim 21, further comprising means for performing a program analysis using the dependence flow graph.

23. (Previously Presented) The method of claim 5, wherein the program analysis further comprises while the set of nodes is not empty iteratively performing:

removing a node from the set of nodes;

if the node is an alias-node, adding the successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement of the form PUT (A, E):

determining a value for E;

updating the initial value based on the value of E; and

adding A to the set of nodes.

24. (Previously Presented) The machine-readable medium of claim 11, wherein the program analysis further comprises while said set of nodes is not empty iteratively performing:

removing a node from the set of nodes;

if the node is an alias-node, adding the successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement of the form PUT (A, E):

determining a value for E;

updating the initial value based on the value of E; and

adding A to the set of nodes.

25. (Previously Presented) The apparatus of claim 17, wherein the operations further comprise:

removing a node from the set of nodes;

if the node is an alias-node, adding the successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement of the form PUT (A, E):

determining a value for E;

updating the initial value based on the value of E; and

adding A to the set of nodes.